International Journal of Future Generation Communication and Networking
Vol. 9, No. 1 (2016), pp. 149-156
http://dx.doi.org/10.14257/ijfgcn.2016.9.1.14

Hybrid Registry for Semantic Web Services based on SWSQT

Haizhong Qian[1] and lili Cai[2]

[1]*College of information technology, jinling institute of technology, Nanjing 211117, China*
[2]*College of media technology, communication university of Nanjing, Nanjing 210017, China*
Hzhqian913@gmail.com lilicai710@163.com

Abstract

The registry of Web Services is a cornerstone for service-oriented architecture and constitutes a critical resource for web services. However, whether the traditional UDDI or its extension based on semantic, has not resolve the issue. The centralized Semantic Web Service Query Tree (SWSQT) is an efficiency service registry which can be dynamically built by service publishing process. The relationships of registered web services will be stored in SWSQT data model so that the large number of ontology reasoning can be avoid at service query state. But the performance of SWSQT will be decreased under the number of services rapidly growth for its centralized architecture. In this paper it proposes a hybrid web service registry based on SWSQT. It can resolve several deficiencies of SWSQT and the experiment has further showed its feasibility and efficiency.

Keywords*: Semantic Web Service, SWSQT, METEOR-S, UDDI, Hybrid*

1. Introduction

Web services technology is rapidly emerging paradigm of distributed architecture and integrating heterogeneous or homogeneous system (or software components) within and across organization boundary. Web services have been widely adopted in e-business、grid computing and sensor web [1] for it can be discovered, invoked over INTERNET. With the rapidly development and proliferation of web services, it is a challenge of organizing web services and quick querying [2, 3]. The registry of web services is a cornerstone for service-oriented architecture and constitutes a critical resource.

UDDI is a standard which is part of web services architecture. But UDDI has crucial limitations and can't satisfy the application requirements. First, it describes functionalities of web services with classification schemes likes NASIC, UNSPSC, etc. The discovery mechanism is based on keywords of classification and the query result would yield coarse results with high precision and recall errors. Second, the data model of UDDI is described by XML. XML descriptions may have very different meanings and vice avers for lacking semantics. Although some approach based on semantic web have been proposed, they also have some limitations, such as low-efficiency, etc. Semantic Web Service Query Tree (SWSQT) is proposed in [4]. SWSQT organize the registered web services by tree model and store their relationships by pre-process. Quick query algorithm based on the tree can reduce the search space and improve the efficiency. However it is a centralized architecture. The performance of SWSQT will be decreased under the number of services rapidly growth. In order to resolve this problem, it proposes a hybrid registry based on SWSQT in this paper.

ISSN: 2233-7857 IJFGCN

The rest of this paper is organized as follows: Section 2 comments on related work briefly; Section3 introduces the SWSQT firstly, and then describes the mechanism of hybrid web services registry; the enhanced quick query web service algorithm is presented in section 4. Section 5 reports and discusses results from the experimentation. Finally section 6 draws some conclusions and talks about possible future directions.

2 Related Works

Web services registries can be classified with regard to their architecture: (a) Centralized, (b) Decentralized, or (c) Hybrid. UDDI is best known centralized registry, but it has some drawbacks for lacking semantic. A lot of effort has been invested in resolving this issue based on semantic web technology. The paper [5] introduces OWL-S into the UDDI standards, and develops a matching engine for web service to accomplish the semantic matching function based on UDDI. Service Operation Tree (SOT) compresses the stored abstract information of web services is presented in [6]. It improves the discovery efficiency, but it lacks semantic supports. A QSQL[7]（Quick Service Query List） mechanism is proposed by Kaijun. Each QSQL node consists of the data domain and link domain. The link domain stores a semantic network structure. The data domain mainly records service information and other required data information. It can avoid time-consuming for ontology reasoning and improve the performance of querying process. SWSQT [4] is a centralized architecture and its querying performance will be decade under the number of services rapidly growth. Schmidt and Parshar [8] present a peer to peer registry architecture based on distributed hash tables, which using an indexing system that is based on the CHORD [9] data lookup protocol. Its curial drawback is that web services are indexed using keywords. The METEOR-S [10] project implements a distributed registry structure. Each registry is mapped to specific domain and classify to certain group by the gateway peer. The METEOR-S has not presented the architecture for storing the web services in local and it will affect its performance of querying web services.

3 Hybrid Registry based on SWSQT

3.1 introduction of SWSQT

SWSQT mainly focuses on the input and output parameters of web services, namely functional characteristics. A web service can be simply described as $ws(I^s, O^s)$, I^s, O^s are respectively, its input parameter and output parameter collection. The structure of SWSQT is tree structure and stores the relationships of registered web services' input parameter collection.

The nodes of the input parameter collection can be considered as non-leaf node of SWSQT, and each web service will be a leaf node belong to the non-leaf node, which if its input parameters contain the collection. Thus, the tree structure can describe the relationships between input parameter collections of web services very intuitively and SWSQT has contains web service's relationship presented in [11].In order to improve the efficiency of querying algorithm, SWSQT also stores the relationships of published web services by special link: such as the exact link, the cause link.

Traditional searching algorithm based on tree structure is not suitable for the performance requirements of SWSQT. SWSQT is a multi-tree, so the efficiency of the traditional traversal search algorithm is bound to be very low. In order to improve the efficiency of query algorithm based on SWSQT, it presents a Query Matrix (QMatrix). The QMatrix is a symmetric matrix and the value of a matrix unit indicates the Clustering Degree between two parameters. The web service querying process locates

the input parameter collection of request on SWSQT by the QMatrix firstly; and then it can locate the non-leaf node of SWSQT; finally, it calculates the web services belong to the node one by one. So the query space will be greatly reduced.

3.2 The Architecture of Hybrid-SWSQT

Although the performance of SWSQT is more efficient among the centralized registries, the QMatrix will increase rapidly under the huge number of web services. In this section, the Hybrid-SWSQT will be presented for resolving the drawback. The architecture of Hybrid-SWSQT is depicted in Figure 1.

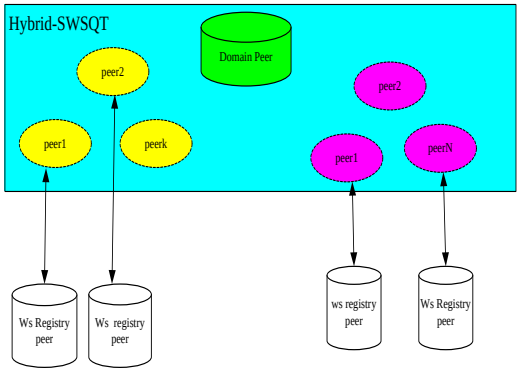

Figure 1. The Architecture of Hybrid-SWSQT

The architecture of Hybrid-SWSQT is a centralized P2P network. The centralized peer manages the access to the peer to peer network for new registry. It also acts as a domain peer, who manages the classification of the domain ontology and the other peers. The other registry peers will be classified some groups by means of the classification of the domain. The registry peers provides transparent registry access through the domain registry. In this way finding web services in a specific domain could be limited to only the registries that are mapped to that domain. When searching for a web service, the registries mapped the specific domain will be searched firstly. And then the other registries will be searched if having no matched web service.

3.3 The Structure of Hybrid-SWSQT

The centralized SWSQT can be expanded to the hybrid web services registry very easily. The structure of SWSQT need to modify is the non-leaf. The non-leaf structure of SWSQT is modified as depicted in Figure2.

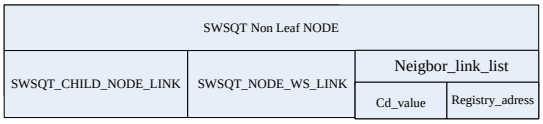

Figure 2. The Node Structure of the Hybrid_SWSQT

The “neighbor_link_list” records those registries, which belong to the specific domain and contain the parameter’s set. The cd pointer is the clustering degree of the query parameters’ set on its local SWSQT. The query process can locate the input parameter collection directly by the pointer, so the performance will be improved for avoiding re-calculate the cluster degree on the local registry.

3.4 The Publishing Web Service Process

The web service publishing process of hybrid-SWSQT is similar in the local registry, except for the updating process in the group. After registering in the local, if the publishing process has created new non_leaf nodes, it will start the updating process in those peers that belong to the

same specific domain. The update information will be sent by broadcast; the other registry in the group receives the updating message and starts the local publishing process. If the new parameters collections exist in the local SWSQT, it will add a record of neighbor_link_list on the exist node, otherwise, it will create new non_leaf nodes.The interaction of publishing web services is depicted in Figure3.

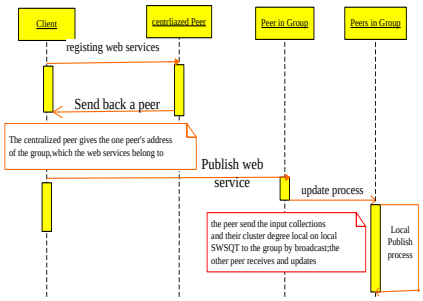

Figure 3. The Interaction of Publishing Web Services

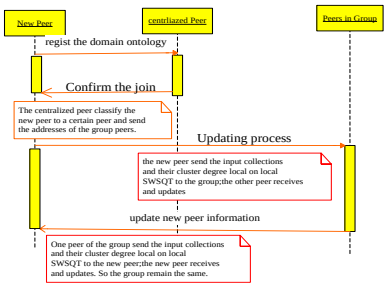

Figure 4. The Interaction of the New Peer Joining

3.5 Interaction of the Initial Process

When a new web service registry wants to join the Hybrid-SWSQT, it will start the initial process (as depicted in Figure 4).First, the new peer accesses the peer to peer network by the centralized peer and registers its' domain ontology in the domain registry; Second it will be classified to a certain groups and receives the lists address of registries in the group; And then it will send the information of local SWSQT to the other peers in the group by broadcast. Those peers receive updating message and start the updating process in local; finally the new peer will start the updating process in the group.

4. Quick Query Algorithm based on Hybrid-SWSQT

According the structure of Hybrid-SWSQT, the query algorithm has three steps: Firstly, it locates the input parameter collection of requirement by means of QMatrix in the local. Because each QMatrix in a group has same information, if the process cannot find a match input parameters collection in the local and it also can't find the other registry of the group, the querying process will be terminated. Otherwise, the querying process will search the specific web services as following sequences:

1) If the point of NODE_WS_LINK is not null, the process search these web services in the local firstly;

2) The process will search theses registries in the same group one by one by means the point of Neighbor_link_list.

The pseudo code of querying algorithm is depicted as follow.

Algorithm ：***QuickQueryWebService based on hybrid-swsqt***

Input: $ws(I_{req}, O_{req})$ //input and result parameter's set of Query;

Output: ws_list //return exact web services

1: if ($I_{req} \equiv \varnothing$) {wslist=get_ws ($hash(f(\varnothing))$;

// get web service adhering the root of local SWSQT, they have no input parameter firstly;
//and then get web service adhering the root of local SWSQT, they have no input parameter;

2: computing similarity between (I_{req}, O_{req}) and ws_i , $ws_i \in wslist$

3: if matched return ws_i }

4:else{m_cd= *GetCDofRequestParameterCollection* (I_{req});

5: if (m_cd==-1) return fails;

6: if (node_ws_link!=NULL){ wslist=get_ws (*hash* (m_cd));// *get web service adhering the node of local SWSQT firstly;*

7: for each $ws_i \in ws\_list$ do {

8:computing similarity between (I_{req}, O_{req}) and ws_i based on semantic respectively

9: end for each;

10: if matched return ws_i }}

11: else { if (neighbor_link!=null) {

12: // *search the other registry in the group one by one just as the local algorithm*

13: return fail;

5. The Primary Experiment and Results

In order to verify the performance of the Hybrid-SWSQT, a test bed is presented in this paper with tools: such as eclipse3.2, JDK1.6, axis2.0, Protege2000 and RACER. And the experiment environments are built on Windows XP, dual-core (1794MHZ), 2.048G memory.

Due to having not a standardized test collection, it creates 3000 web services randomly using the concepts of ontologies[1] as inputs and outputs parameters.

For a more objective compare in the experiment, Hybrid-SWSQT will be tested with another three methods, namely, direct ontology reasoning, Quick Services Query List (QSQL) and METEOR-S. The experiment will be tested in two conditions as follow. First test is analyzing the performances of these centralized registries. The three methods will be test several times under the different amount of the web services. The result is depicted in Figure5. The second experiment is analyzing the performance between hybrid-SWSQT and METEOR-S. The result is depicted in Figure6.

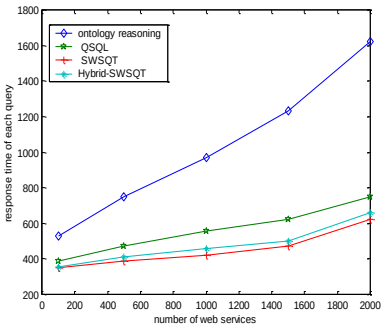

Figure 5. The Performance of These Centralized Registries

[1] http://protegewiki.stanford.edu/index.php/Protege ontology library

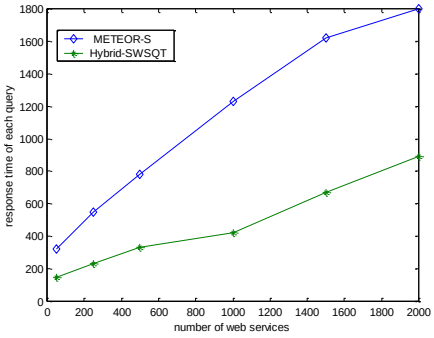

Figure 6. The Performance between Hybrid-SWSQT and Meteor-S

As can be seen from Figure 5, although the architecture of Hybrid-SWSQT is hybrid, its performance is the same as the centralized SWSQT and better than the other registries. Because each SWSQT in the same group remains the same through the updating mechanism and the cluster degree of each input parameter is also recorded avoiding re-calculate, the difference of time consuming of process between SWSQT and Hybrid-SWSQT is very little. QSQL constructs a chain of concepts' relationship and avoids the process of ontology reasoning. Although it improves effect of query algorithm to some extent, QSQL has not reduced the query space effectively. The algorithm of direct ontology reasoning has the worst effect in that it needs to do the ontology reasoning algorithm for each service one by one and consumes more time.

Figure 6 shows that the effect of Hybrid-SWSQT among the hybrid registries is also obvious. Due to the relationship among all the services will be built in local SWSQT. The query process of Hybrid-SWSQT will be limited in specific registries just as in the local. The METEOR-S has not established an efficient structure for organizing those registered web services and recorded their relationships before querying.

The experimental results show that Hybrid-SWSQT proposed in this paper has high discovery efficiency, and can provide important support for method of semantic web service composition, which can be performed in the dynamical runtime.

6 Conclusions

The quick selection of suited web services from adequate registries is important at the design time of web service composed systems, especially composing web services in the dynamical runtime. Although many approaches have been presented in past years, the performance of these approaches can't meet the requirement. SWSQT is a perfect registry among the centralized registries, but the performance will be reduced under the huge numbers of registered web services. So the hybrid web services registry based on SWSQT is proposed in this paper.

The hybrid-SWSQT not only retains the advantages of centralized SWSQT, but also it classifies the registries to some groups by domain ontology. The querying process will be limited to some specific registries and reduces the time consuming. The other advantage of hybrid-SWSQT is that their QMATRIX in a group will remain the same by updating process. It means the input parameters of requirement only be calculated once and improves the performance of query process. The updating message just contains the input parameters and it will reduce the network bandwidth. The updating process will be limited in those specific registries, which can avoid the broadcasting storm.

The current work of Hybrid-SWSQT focuses the functional properties of web services and the non-functional properties will be considered in the future.

Acknowledgements

This works has been supported by the Natural Science Foundation of Colleges and universities in Jiangsu Province (Grant No.14KJB520013) and The Doctoral Fund of jinling institute of technology. (No. jit-b-201324, No. 2014-jit-n-5).

References

[1] Sheth, C. Henson and SS. Sahoo, "Semantic sensor web", *IEEE internet computing*, vol. 12, no. 4, **(2008)**, pp. 78-83.
[2] Zeng and K.Q. He, "Toward multi-ontology based interoperability in web service registry", *Journal of Computational Information Systems*, vol. 5, no. 6, **(2009)**, pp. 1669-1677.
[3] H. Yang, "A Semantic Matching Algorithm of Web Services Based on Google Distance", *Journal of Computational Information Systems*, vol. 7, no. 8, **(2011)**, pp. 2624- 2633.
[4] H. Qian and S. Subin, "SWSQT(Semantic Web Service Query Tree): Quick Discovery and Organization Mechanism for Web Services", *Journal of Computational Information Systems*, vol. 7, no. 11, **(2011)**, pp. 4015-4022.
[5] N. Srinivasan and M. Paolucci, "an efficient algorithm for OWL-S based semantic search in UDDI", *Semantic Web Services and Web Process Composition*, Springer Berlin / Heidelberg, **(2005)**, pp. 96-110.
[6] D.A. D'Mello and V.S. Ananthanarayana, "A Tree Structure for Efficient Web Service Discovery", *Second International Conference on Emerging Trends in Engineering & Technology*, **(2009)**, pp. 826-831.
[7] R. Kaijun, C. Jinjun, X. Nong and S. Junqiang, "building quick service query list(QSQL) to support automated service discovery for scientific workflow", *concurrency and computation: practice and experience*, vol. 21, **(2009)**, pp. 2099-2117.
[8] Schmidt and M. Parashar, "A peer-to-peer approach to Web services discovery", *Proceedings of the 2003 International Conference on Web Services(ICWS'03)*, **(2003)**.
[9] Y. Stavrakas and M. Gergatsoulis, "Multidimensional semistructured data: Representing context-dependent information on the Web", *In Proc. of the 14th Int. Conf. on Advanced Information Systems Engineering(CAISE'02)*, Toronto, Canada, **(2002)**.
[10] K. Verma and K. Sivashanmugam, "METEOR-S WSDI: Ascalable P2P infrastructure of registries for semantic publication and discovery of Web services", *Journal of Information Technology and Management*, **(2004)**, pp. 1-24.
[11] M. Paolucci and T. Kawamura, "Semantic Matching of web Services Capabilities", *Proceedings of the first international semantic web conference on the semantic web*, **(2002)**, pp. 333-347.

Authors

Haizhong Qian (1997-09-13), Huangshan City, China
He is a phD. His research interestes include semantic web, semantic web service and sensor web. Contact him at the jinling institue of technology, nanjing, china;

Lili Cai (1977-07-10) Nanchan City, China
She is a PhD student at nuaa. His research instrest include sigal processing. Contact her:lilicai710@163.com.